From Statistical Model Checking to Statistical Model Inference: Characterizing the Effect of Process Variations in Analog Circuits

Yan Zhang[1], Sriram Sankaranarayanan[1], Fabio Somenzi[1], Xin Chen[2] and Erika Ábraham[2]
[1]University of Colorado at Boulder, Boulder, CO, USA.
[2]RWTH Aachen University, Aachen, Germany.
Email: [1]{yan.zhang,srirams,fabio}@colorado.edu, [2]{xin.chen,abraham}@informatik.rwth-aachen.de

***Abstract*—This paper studies the effect of parameter variation on the behavior of analog circuits at the transistor (netlist) level. It is well known that variation in key circuit parameters can often adversely impact the correctness and performance of analog circuits during fabrication. An important problem lies in characterizing a *safe subset* of the parameter space for which the circuit can be guaranteed to satisfy the design specification. Due to the sheer size and complexity of analog circuits, a formal approach to the problem remains out of reach, especially at the transistor level. Therefore, we present a statistical model inference approach that exploits recent advances in statistical verification techniques. Our approach uses extensive circuit simulations to infer polynomials that approximate the behavior of a circuit. A procedure inspired by *statistical model checking* is then introduced to produce "statistically sound" models that extend the polynomial approximation. The resulting model can be viewed as a statistically guaranteed over-approximation of the circuit behavior. The proposed technique is demonstrated with two case studies in which it identifies subsets of parameters that satisfy the design specifications.**

I. Introduction

In this paper, we address the problem of estimating a safe region of the parameter space that guarantees a design specification of an analog circuit. The parameters considered by our approach, such as capacitance, channel width of transistors and thickness of the oxide layer, originate from process variations. While circuits are commonly designed by assuming nominal values of these parameters, it is often desirable to examine the effect of process variations on important design specifications. In this paper, we examine specifications that are of the form $\phi(\vec{p}) \in [\ell, u]$ where ϕ is a performance metric and $\vec{p}$ is a vector of process parameters. The choice of ϕ includes a wide variety of properties ranging from time-bounded temporal logic properties to those that are not easily expressible in temporal logic, such as the oscillation frequency of an oscillator, the common mode rejection ratio (CMRR) and the slew rate of an operational amplifier. Our approach identifies a set of "safe" parameter values that satisfy the design specifications with high confidence.

While definite progress has been achieved using formal verification approaches to prove properties of analog circuits, the state-of-the-art falls short of addressing analog circuits at the transistor level [1]. Therefore, we explore statistical approaches that rely on extensive simulations of the circuit using a standard simulator such as SPICE. Our approach is based on two main ideas: (1) We use simulation and regression to fit a polynomial (or piecewise polynomial) approximation q for a property ϕ; (2) We present a statistical hypothesis testing approach to finding a *bloat interval* I so that $q \oplus I$ is guaranteed to over-approximate ϕ for a given fraction θ of the parameter space with high confidence. The interval I "bloats" the model q to make it *statistically sound*. Our approach to finding this interval has been inspired by recent developments in the theory of statistical model checking (SMC) [2]–[4]. Whereas the SMC approaches answer a yes/no verification question given a circuit and a design specification, we exploit the key idea of using hypothesis testing to find tolerance intervals over an approximation. The final model (q, I) constructed by our approach can be used to characterize a set of parameter values that are statistically guaranteed to satisfy the specification with high confidence. Such a characterization is already beyond the realm of existing statistical model checkers.

We implemented our approach in a prototype tool that uses an open-source circuit simulator, ngspice [5]. We demonstrate our approach over several benchmark circuits. We show the usefulness of our approach in identifying a region of the parameter space that satisfies the specification with high confidence. Finally, we compare our approach against standard Monte-Carlo (MC) simulations and statistical model checking in terms of the verification ability and the number of simulations required. While our approach seeks to discover information about circuits that is out of the scope of either technique, we observe that the overhead, in term of either the computational cost or the simulation cost, on top of standard MC simulations and SMC is quite reasonable.

A. Related Work

A large volume of work exists on analog circuits verification, ranging from testing to formal verification. We restrict our discussion to techniques that are closely related.

Formal verification of analog circuits dates back to the work of Kurshan *et al.* [6] and Hedrich *et al.* [7]. One of the most important approaches is to perform reachability analysis on a discretized state-space that approximates the original continuous, infinite state-space. Frehse *et al.* [8] proposed a forward/backward abstraction refinement scheme for reachability.

This work was partially supported by the US National Science Foundation (NSF) under the award number 1016994. All opinions expressed are those of the authors, and not necessarily of the NSF.

978-1-4799-1071-7/13/$31.00 ©2013 IEEE

The approach produces accurate results, but is computationally expensive even for small circuits. In the series of papers by Myers *et al.* [9], [10], labeled hybrid Petri nets were used to formalize and model check circuits. However, their techniques mostly focus on the behavioral models of analog circuits. In the work of Tiwary [11] SMT solvers were used to reason about piecewise linearized circuits. As shown in [1], this technique does not scale well with the size and complexity of circuits. Althoff *et al.* [12] proposed a continuation technique for the reachability analysis of phase-lock loops (PLLs). While their result is certainly an exciting progress, the approach verifies behavioral models rather than a transistor-level implementation. The technique for PLL verification proposed by Yin *et al.* [13] has the same issue. In [14] an interesting proof about the start-up of an even-stage ring oscillator was presented. The authors also proposed a variable substitution technique to lower the state-space dimension. A key shortcoming of existing formal approaches lies in the inability to reason about circuits with even a few transistors (e.g., less than 10) without resorting to simplified device models in the verification process.

Another thread of research uses repeated simulations enriched with statistical hypothesis testing techniques to provide statistical guarantees on circuits. While the underlying theory is well known [15], these approaches have not been well-recognized by the verification community until recently. Younes and Simons [2] proposed a technique called statistical model checking (SMC). They regard the model checking of stochastic systems as a hypothesis testing problem and solve it using sequential probability ratio test (SPRT) [15]. Later, Sen *et al.* proposed a p-value significance test [16] for the verification of black-box systems. Zuliani *et al.* [4] proposed a Bayesian estimation approach. Their approach computes an interval estimate for the probability of satisfying a bounded LTL (BLTL) formula. Jha *et al.* [3] introduced a new SMC framework based on Bayesian hypothesis testing [17], [18]. Compared with SPRT, Bayesian hypothesis test is more convenient since it does not require indifference regions. Instead, it computes Bayes factor by computing integrals over a given prior density. Recently, Bayesian SMC was applied to the verification of an operational amplifier [19].

An important difference between our approach and the existing statistical techniques is that we perform *model inference* as opposed to verification. The resulting models provide useful information to help with improving circuit design, which cannot be achieved by existing techniques [2]–[4], [19]. A key contribution of this paper is to establish the model inference procedure under the same framework of SMC and utilize the insights from SMC to carry out the inference.

The use of MC and Quasi MC simulations is quite standard for the analysis of analog circuits. The approach of Singhee and Rutenbar [20] used ideas from extreme value theory to estimate bounds on the timings of analog circuits. The idea is to perform random simulations while observing excesses of previously seen timing levels. Then a generalized Pareto distribution to these excesses is fit and used to predict values of the extremal timings of the circuit. We are also interested in extremal deviations of circuit metrics from a polynomial approximation fitted through regression. However, instead of using extreme value theory, we use a statistical hypothesis testing approach. Comparing both approaches in terms of their strengths and weaknesses will form an important aspect of our future work. Besides extreme value theory, Singhee *et al.* also used an interesting idea of *statistical blockade* to avoid unnecessary simulations and thus sample efficiently from the tail of the input distribution. This is another potential interesting idea to explore in the setting of this paper.

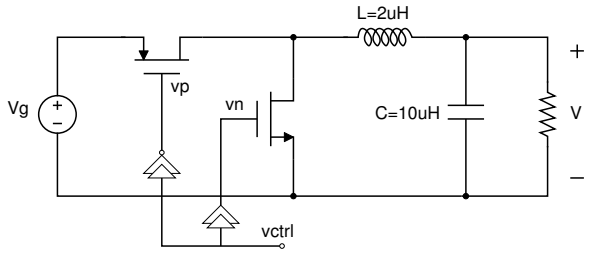


Fig. 1: A buck converter with L and C taking values from $[1.8, 2.2]\mu$H and $[9, 11]\mu$F respectively.

It is also interesting to characterize the distribution of state variables over time, which arises from the uncertainties in a circuit. This can be tackled by techniques for uncertainty quantification, such as the polynomial chaos expansion (PCE) [21]. The theory of PCE was applied by Strunz and Su [22] to build a simulator for stochastic simulation of analog circuits. Since the objective of this paper is to study how design properties are affected by process variations, deriving accurate distributions on the state variables over time may not be necessary.

Besides statistical approaches, there are other interesting techniques originating from the hardware testing community. Yoon *et al.* [23] proposed a hierarchical model inference approach to derive statistical distributions of circuit properties. Dang *et al.* [24] used motion planning techniques for rapidly-exploring random trees (RRTs) to verify specifications of analog circuits. Ahmadyan *et al.* [25] also used RRTs to generate property-oriented test cases for analog circuits.

II. Overview

In this section, *statistical model inference* (*SMI*), is presented with a running example. We illustrate the basic flow of this technique and defer the details to a later section.

A. Problem Setup

Let $\vec{p}$ be the process parameters in a circuit and $\mathcal{P}$ be a parameter space. The joint distribution $\mathcal{D}$ of the parameters is assumed. We present the problem setup by a running example on a buck converter. The assumption of independence and uniform distribution in the example are purely for illustration. In principle, SMI places no restriction on the underlying distribution as long as samples can be drawn accordingly.

Example II.1 (Buck Converter)**.** *Figure 1 shows a buck converter. The nominal values of the inductor L and the capacitor C are 2μH and 10μF, respectively. We assume that L and C are independent random variables uniformly distributed in the range $L \in [1.8, 2.2]\mu$H and $C \in [9, 11]\mu$F.*

Assume that $\vec{x}(t|\vec{p})$ is a set of circuit state variables, such as currents and voltages, over a time interval $[0, T]$. Given a set of

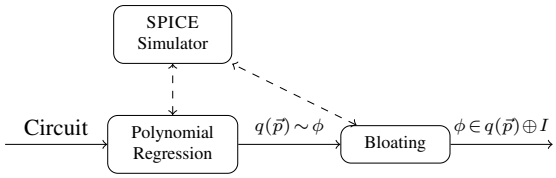


Fig. 2: A diagram showing the flow of SMI.

process parameters, a property ϕ is a *performance metric* that maps $\vec{x}(t|\vec{p})$ to a real value. SMI works with a property that can be written as a function $\phi(\vec{p})$ that depends on the process parameters. In practice, ϕ is evaluated by simulating the circuit with specific parameter values, computing the variables $\vec{x}$ over time and measuring the trajectories $\vec{x}(t)$ properly. We consider design specifications of the form $\phi(\vec{p}) \in [\ell, u]$ where ℓ, u are the tolerance limits for ϕ.

Example II.2. *We choose the* ripple amplitude Δv *as a property of interest in the buck converter example. For the circuit in Figure 1,*

$$\Delta v = \phi(L, C) = \frac{V_g - V}{16LC} D T_s^2 \,, \tag{1}$$

where V is the DC component of the output voltage, D is the duty cycle and T_s is the time period of the control voltage. We assume that $V = 3V$, $D = 0.25$ and $T_s = 2\mu s$.

We are interested in choosing L and C such that $\Delta v \leq 30mV$. Equation (1) *yields $LC \geq 18.75$. However, such an analytic solution is only possible for simple circuits since ϕ generally has a complicated dependence on $\vec{p}$. Therefore, we seek approaches to approximate ϕ in terms of $\vec{p}$.*

Let q be a polynomial of a given degree d, and I an interval. SMI approximates a property ϕ by showing

$$\phi(\vec{p}) \in q(\vec{p}) \oplus I \,,$$

where $\oplus$ denotes the Minkowski sum. The interval I (called the bloat) is found by a statistical hypothesis testing procedure explained in the following. Given (q, I), we wish to identify a set $S \subseteq \mathcal{P}$ such that for all $\vec{p} \in S, q(\vec{p}) \oplus I \subseteq [\ell, u]$.

B. Overview of Statistical Model Inference

Figure 2 shows a diagram for the flow of SMI. We assume that the circuit under consideration can be simulated efficiently using a standard simulator. SMI employs two phases: (1) a regression phase in which we fit a polynomial q, and (2) a bloating phase in which we generate a bloat interval I.

Regression. A number of $N > 0$ samples, $\vec{p}_1, \ldots, \vec{p}_N$, distributed according to $\mathcal{D}$, are drawn from the parameter space $\mathcal{P}$. The circuit is simulated for each $\vec{p}_i$ and the value $\phi(\vec{p}_i)$ is measured. We use least squares to fit a polynomial $q(\vec{p})$ of degree d such that the overall $\mathcal{L}_2$ error is minimized:

$$q = \arg\min \sum_{i=1}^{N} \left(\phi(\vec{p}_i) - q(\vec{p}_i)\right)^2 .$$

The polynomial q is only an approximation of ϕ. Few guarantees can be placed on it. The core of our technique is a model inference technique that uses ideas from statistical model checking, namely the sequential hypothesis testing.

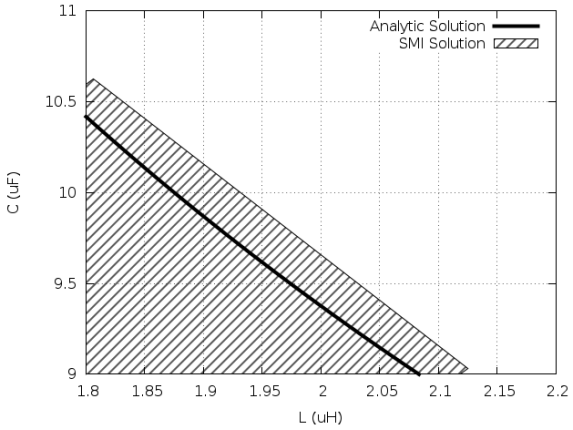


Fig. 3: Verification result of SMI on a buck converter. The region below the solid curve is known to be unsafe. The shaded region is the unsafe subset predicted by SMI.

Bloating. Let $\theta \in (0, 1)$ be a user-specified probability. Given a polynomial q, we wish to find an interval I such that with a probability of at least θ, we have $\phi(\vec{p}) \in q(\vec{p}) \oplus I$ for a randomly chosen $\vec{p} \in \mathcal{P}$. Typically, θ is chosen close to 1.

In SMI, the choice of I is treated as a hypothesis testing problem. Given two mutually exclusive hypotheses:

$$\mathcal{H}_0 : \mathsf{Pr}(\phi \in q \oplus I) \geq \theta \text{ and } \mathcal{H}_1 : \mathsf{Pr}(\phi \in q \oplus I) < \theta \,,$$

a standard statistical hypothesis testing procedure accepts $\mathcal{H}_0$ over $\mathcal{H}_1$ if I is considered large enough. Later in this paper, we show a simple yet elegant variation of the standard tests such as SPRT and the Jeffreys' test, which allows us to find such an interval I on the fly.

Identification of safe parameters. The regression and bloating steps effectively create a model (q, I) of the target property ϕ such that a statistical model checker can be convinced that $\phi \in q \oplus I$ with a given probability θ. Given such a model, identifying the safety of the parameter values $\vec{p}$ reduces to checking whether $q(\vec{p}) \oplus I \subseteq [\ell, u]$ holds. Furthermore, S, the safe subset of the parameter space, can be explicitly computed by interval arithmetic in a conservative manner.

Example II.3. *Back to the buck converter example, we perform $N = 20$ simulations to produce a polynomial $q_3(L, C)$ of degree 3, as shown below, to approximate the ripple Δv.*

$$\left(\begin{array}{c} 0.17 - 1.12 \times 10^4 C + 2.80 \times 10^8 C^2 + \\ 8.48 \times 10^3 C^3 - 5.64 \times 10^4 L + 1.40 \times 10^9 CL + 3.01 \times 10^4 C^2 L + \\ 7.05 \times 10^9 L^2 + 7.66 \times 10^4 CL^2 + 4.23 \times 10^4 L^3 \end{array} \right) .$$

Next, q_3 is bloated using $\theta = 0.95$. The resulting interval is $I = [-37, 44]\mu V$. The model (q_3, I) is used to check the design specification. $\Delta v \leq 30mV$. The result is shown in Figure 3. The solid curve plots of the equation $LC = 18.75$, and thus the region below is known to be unsafe. The shaded region is the unsafe subset predicted by SMI.

III. Statistical Model Inference

The two phases of SMI are briefly presented in the previous section. Now we discuss issues that arise in the two phases and give a detailed algorithm for model inference. Throughout this section, we assume that ϕ, a design property, is a continuous function of the parameters $\vec{p}$. The assumption is reasonable since analog circuits are usually modeled as continuous systems that exhibit continuous dependence on the system parameters. Note that in some cases, ϕ is not necessarily a function. But as we show later, this does not restrict the application of SMI.

A. Polynomial Approximation

It is well known that any continuous function over a bounded domain can be approximated "arbitrarily closely" by a polynomial. In practice, we are interested in polynomial approximations with a fixed degree. There are many ways to formulate such an approximation, such as Chebyshev and Bernstein polynomials. In our case, we consider a simple scheme based on least squares fitting.

Let $\phi(\vec{p})$ be a continuous function and d be the degree of a polynomial. Formally, we want to find a polynomial

$$q = \arg\min \int_{\mathcal{P}} (\phi(\vec{p}) - q(\vec{p}))^2 d\mu(\vec{p}) \,.$$

However, integrating over all $\vec{p} \in \mathcal{P}$ is often hard if not impossible. As an alternative, we instead compute

$$q = \arg\min \sum_{i=1}^{N} (\phi(\vec{p}_i) - q(\vec{p}_i))^2 \,,$$

where $\vec{p}_i, i = 1, \ldots, N$ are samples drawn from $\mathcal{P}$ according to a certain distribution $\mathcal{D}$.

Choosing the number of samples. One question that arises in practice is the number of samples, N, required to find a "good" polynomial fit. If N is too small, q may be an artifact of the samples rather than of the property ϕ itself. On the other hand, a large N incurs overheads in simulation as well as in the least squares regression.

We present a simple heuristic called *resampling*, which is inspired by k-fold cross validation. For a set of N samples, we partition it into k folds (typically $k \in [4, 10]$) of $\frac{N}{k}$ samples. We create k different sets, each with $\frac{k-1}{k}N$ samples, such that each fold is left out exactly once. A set of polynomials $q^{(i)}, i = 1, \ldots, k$ are constructed by least squares fitting on each set of samples. The number of samples N is considered adequate if the differences between the polynomial coefficients are within some tolerance. In this case, we obtain $q = \frac{1}{k}\sum_{i=1}^{k} q^{(i)}$. If N is judged not enough, we draw more samples and repeat the above procedure. Clearly, the resampling technique provides a systematic way to determine the sample size and reduces the bias of the resulting approximation.

The computational complexity of finding a polynomial approximation is dominated by least squares fitting which is known to be $O(NV^2)$, with N the number of samples and V the number of unknown coefficients. Therefore, as the number of variations grows, least squares regression can become inefficient. Employing more sophisticated regression algorithms is important in the future work.

Data: Parameter Space $\mathbb{P}$, Polynomial q, Run Length K
Result: Bloat Interval I
$\mathsf{count} := 0$;
$I := [0, 0]$;
while $\mathsf{count} < K$ **do**
  $\vec{p} := \mathsf{do\_random\_sampling}(\mathbb{P})$;
  $\phi := \mathsf{simulate\_and\_measure\_property}(\vec{p})$;
  if $\phi \notin q(\vec{p}) \oplus I$ **then**
    $\mathsf{count} := 0$;
    $I := \mathsf{update\_interval}(I, \phi - q(\vec{p}))$;
  else
    $\mathsf{count} := \mathsf{count} + 1$;
  end
end

Algorithm 1: Deriving a bloat interval I by observing K consecutive successes.

B. Bloating via Hypothesis Testing

Now consider the problem of finding the bloat I given a probability θ. As mentioned earlier, given a bloat I, we can use hypothesis testing techniques to determine whether it is good enough. The two mutually exclusive hypotheses

$$\mathcal{H}_0 : \mathsf{Pr}(\phi \in q \oplus I) \geq \theta \text{ and } \mathcal{H}_1 : \mathsf{Pr}(\phi \in q \oplus I) < \theta \,.$$

are known as the null and the alternative hypothesis. Such a technique uses repeated simulations to decide between the two hypotheses. In our case, we choose to use the Bayesian test, which proceeds by computing the Bayes factor [18]

$$\mathcal{B} = \frac{\mathsf{Pr}(d|\mathcal{H}_0)}{\mathsf{Pr}(d|\mathcal{H}_1)} \,,$$

where d is a collection of Bernoulli random variables denoting the outcome of the inclusion test $\phi(\vec{p}_i) \in q(\vec{p}_i) \oplus I$ with random samples $\vec{p}_i$. If $\mathcal{B}$ becomes larger than a given threshold T, $\mathcal{H}_0$ is accepted. The threshold T is known as the *Bayes factor threshold*. Bayesian test also assumes that θ, which can be regarded as a random variable ranging from $(0, 1)$, distributes according to a *prior* density. In practice, when no knowledge is available, we often assume a uniform prior for θ.

The goal of SMI, however, is to discover such a bloat interval. We adopt a procedure based on hypothesis testing to achieve this, which is summarized in Algorithm 1. Denoting a bloat $I = [I_l, I_u]$, the procedure starts with a zero bloat such that $I_l = I_u = 0$. It then repeatedly checks the formula $\phi(\vec{p}_i) \in q(\vec{p}_i) \oplus I$ with samples $\vec{p}_i$ drawn from the parameter space $\mathcal{P}$ according to the distribution $\mathcal{D}$. We say that the inclusion test is a success if the above formula holds, and a failure otherwise. Upon each success, we increment the counter and continue with fresh samples until a failure occurs. In this case, we update the bloat $I := [\min(r, I_l), \max(r, I_u)]$ with $r = \phi(\vec{p}_i) - q(\vec{p}_i)$, to include the sample $\vec{p}_i$ that causes the failure. Also, the counter is reset to zero. The procedure terminates when a run of $K > 0$ consecutive successes is seen.

Deriving K for Bayesian test. A standard Bayesian hypothesis testing procedure does not compute K since the computation of the Bayes factor effectively serves the same purpose. In SMI, we are only interested in consecutive successes since those samples that fail the inclusion tests are used to update the interval I. This simplifies the computation of the Bayes factor. Furthermore, for a given θ and T, we can derive a

proper length K such that a run of K consecutive successes is sufficient for the Bayesian test to accept $\mathcal{H}_0$ over $\mathcal{H}_1$. Here we show a simple case that θ has a uniform prior. In practice, K can be computed for any given prior.

Given an observed sequence of n consecutive successes, the corresponding Bayes factor is given by

$$\mathcal{B} = \frac{\int_\theta^1 x^n dx}{\int_0^\theta x^n dx} = \frac{1-\theta^{n+1}}{\theta^{n+1}}\,.$$

Since $\mathcal{H}_0$ is accepted as soon as $\mathcal{B} \geq T$, we have

$$K \geq -\frac{\log T + 1}{\log \theta} - 1\,.$$

Therefore, once we observe a run of K consecutive successes, we can stop the bloating. Table I shows some of the values for K given the desired probability θ and threshold T. Increasing θ and T requires a larger K, leading to a model with better statistical guarantee. In practice, we find that $\theta = 0.95$ and $T = 100$ often provide a good trade-off between statistical guarantee and computational cost.

TABLE I: Run length K for common values of θ and T.

	T = 100	500	1000
θ = 0.9	43	59	65
0.95	89	121	134
0.99	459	618	687

IV. Case Study

In this section, two case studies on the application of SMI are presented. The first one is on a three-stage *ring oscillator*. We use SMI to verify that the oscillation frequency is within a specified range. The other case is on a two-stage *operational amplifier* (*opamp*). Several interesting design specifications of this circuit are verified. We use an open source program, ngspice (revision 25) [5], as the circuit simulator. All the experimental results are measured on a quad-core 2.8GHz machine running Debian 6.0.

A. Ring Oscillator

A three-stage ring oscillator is shown in Figure 4. The circuit is designed to work at a frequency of 3.85GHz with a tolerance of $\pm$50MHz. To find the oscillation frequency, we use the periodic steady-state (PSS) analysis in ngspice. The oscillation frequency of this circuit is determined by various process parameters of each transistor $M_i, i = 1, \ldots, 6$. In our case, we choose to fix other parameters and let the channel width w_i of each transistor have 5% variation uniformly distributed around the nominal value.

The verification builds a statistically sound model of the oscillation frequency with respect to the 6 parameters, with $\theta = 0.95$ and $T = 100$. The polynomial in the model is of degree 3. Table II shows the number of simulations required for the first and the second phase of SMI and the corresponding times. The column "#SIMS" shows the number of simulations. The column "SIM-TIME" indicates the simulation time. "OTHER-TIME" refers to the time cost of the computation each phase besides simulation, i.e., resampling and least squares fitting in the first phase and bloating in the second phase.

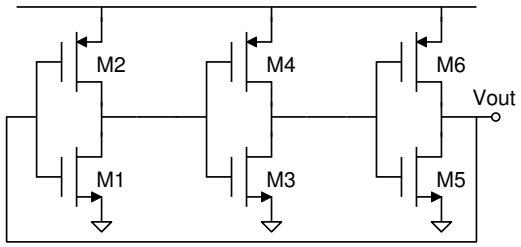


Fig. 4: A three-stage ring oscillator.

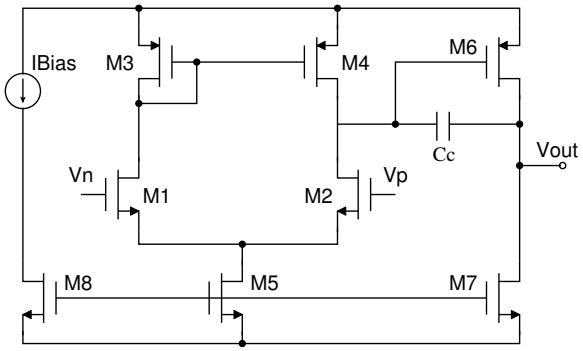


Fig. 5: A two-stage operational amplifier.

SMI computes a bloat $I = [-19, 23]$MHz. It is verified that the oscillation frequency is within the range 3.85GHz$\pm$50MHz with the given process variations.

TABLE II: The number of simulations and computation times for the oscillator verification.

	#SIMS	SIM-TIME	OTHER-TIME
Phase 1	150	725s	3s
Phase 2	265	1280s	0.4s

B. Two-Stage Operational Amplifier

Opamps are key components in many analog circuits. They often have two inputs, a positive and a negative power supply, and an output. An opamp can serve two purposes. The first is to amplify small input signals. In this case, the opamp works in linear mode such that the output approximately equals the difference between the two input signals multiplied by the gain of the opamp. The other mode of operation is as a comparator in which the output is driven close to either the positive or the negative supply voltage, depending on which input is larger.

Experimental setup. The schematic of a commonly used two-stage opamp is shown in Figure 5. The performance of an opamp is characterized by many properties, as shown in Table III. Usually, each property is measured using a specific type of simulation and circuit configuration. For example, the input offset voltage is often measured by arranging the opamp in the unity-gain configuration and sweeping DC input voltage. On the contrast, the measurement of the slew rate requires transient simulation. In Table III, we also list the type of simulation for each property. For a detailed description on how to simulate these properties, we refer the interested reader to [26] (Chapter 6.6).

We select 8 process parameters to study, including the oxide thickness t_{ox}, threshold voltage under zero bias v_{t0}, channel width w and channel length l for the transistors M_1 and M_2. To distinguish between parameters for different transistors, we add subscripts $i = 1, 2$ at the end of the parameter names. All the 8 parameters are assumed to be uniform and have 5% variations around nominal values.

Table III summarizes the set of properties that we are interested in. Table IV summarizes the set of parameters, the type and degree of polynomial for each property. We now provide a detailed rationale for these choices.

Choosing parameters. A common object of study is the mismatch in the transistor parameters between M_1 and M_2 in Figure 5. Therefore, rather than treating parameters p_1 and p_2 separately, it may be more interesting to study the performance metric directly as a function of $p_1 - p_2$. However, this needs to be considered on a case-by-case basis. For instance, Figure 6a shows the unity-gain bandwidth as a function of w_1 and w_2 (the transistor widths) separately while Figure 6b shows it as a function of $w_1 - w_2$. It can be observed that the bandwidth has a nearly linear dependence on w_1 and w_2. However, it is hard to find a simple relationship between the bandwidth and $w_1 - w_2$. In practice, the latter choice (using $w_1 - w_2$) provides a poor model with a large bloating interval. In such a situation, we use w_1 and w_2 as the parameters in our SMI algorithm.

By contrast, the data for the DC voltage gain are shown in Figure 7. In this case, the relationship between $w_1 - w_2$ (the width mismatch) and the DC voltage gain is quite clear. Note that the relationship is not functional. But this does not restrict us from applying the regression algorithm. In practice, we observe that in SMI, using $w_1 - w_2$ as the parameter provides us a simple polynomial approximation with a small bloat interval. Also, much fewer simulations are required to fit this polynomial than the case for using w_1 and w_2 as separate parameters. Therefore, it is clear that a careful choice of parameters is essential to generate good models with as few simulations as possible.

Choosing the type of polynomials. Choosing the degree of the approximations is yet another challenge. In some cases, using piecewise polynomial approximations is beneficial, allowing us to fit polynomials with smaller degrees. Figure 8 shows a situation involving the DC voltage gain. It clearly shows that a degree 3 piecewise polynomial approximation with 2 pieces outperforms larger degree approximations in terms of the approximation error. Also, the generation of the piecewise polynomial requires fewer number of simulations.

In general, we may use powerful machine learning techniques such as random forests to learn piecewise approximations. In SMI, however, we use a simple visual plot as in Figure 8 to determine the number of pieces and the points of discontinuity in our regression routine. Integrating our work with more sophisticated piecewise polynomial regression will be considered in the future.

Experimental results. The experimental results are summarized in Table V under the column "SMI". For each property, a statistically sound model is constructed using $\theta = 0.95$ and $T = 100$. The columns "#SIMS-1" and "#SIMS-2" show the numbers of simulations and the total simulation time in the first and the second phase of SMI, respectively. The column

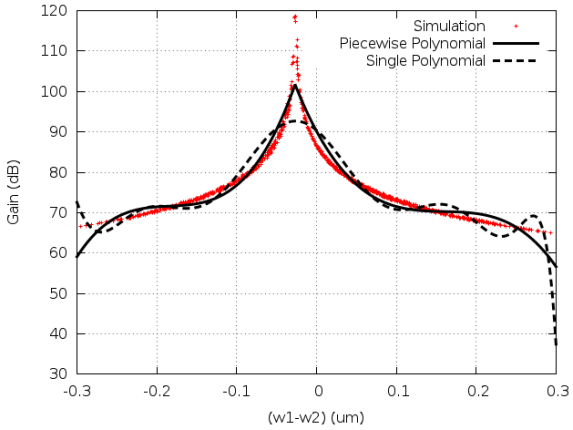


Fig. 8: Comparison between a 2-piece third-order polynomial and a degree 10 polynomial fit for the same data.

TABLE IV: The set of parameters and the type of polynomial for each property. The column "#PIECES" indicates the number of pieces used (1 indicates that a single polynomial was used). The degree of polynomial is fixed to 3 for all the properties.

ID	PARAMETERS	# PIECES
1	$t_{ox1} - t_{ox2}$, $v_{t01} - v_{t02}$, $w_1 - w_2$, $l_1 - l_2$	1
2		4
3	$t_{ox1} - t_{ox2}$, $v_{t01} - v_{t02}$, w_1, w_2, l_1, l_2	1
4		1
5		1
6		1
7		1
8	t_{ox1}, t_{ox2}, v_{t01}, v_{t02}, w_1, w_2, l_1, l_2	1
9		1

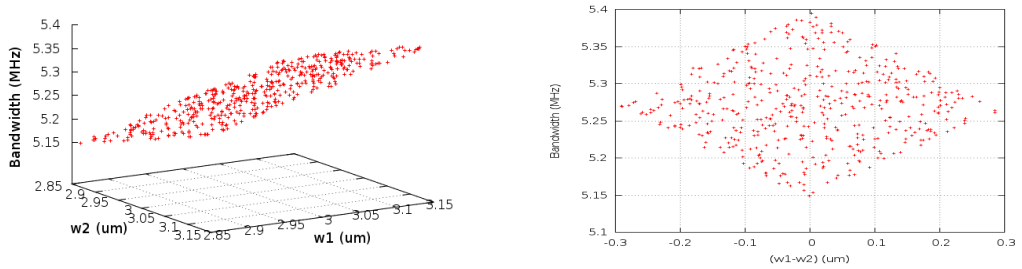
"TIME-1" shows the time spent on resampling and least squares fitting. The column "TIME-2" shows the time spent on the computation of bloat intervals. The verification results are shown in the last column. If a property fails to satisfy the corresponding specification, we compute a safe subset of the parameters using the inferred model. The table reports the probability of satisfaction by computing the area of this safe subset. For comparison, we also include the results for 1000 Monte-Carlo (MC) simulations and the Bayesian statistical model checking (SMC) with $\theta = 0.95$ and $T = 100$.

First, we compare SMI with the MC simulations. Observe that the verification results of the two approaches are the same in the sense that SMI correctly shows which specification is satisfied and which is not. For the two unsatisfied specifications, the probabilities of satisfaction predicted by SMI are lower than the yield of MC simulations. This is because SMI provides under-approximations of the safe subsets.

Next, SMI is compared with SMC. We observe that SMI generally needs more simulations than SMC, especially when a specification is satisfied. It is because SMI is not just a verification technique but also does model inference. A key difference between SMI and SMC is that SMI can show which part of the parameter space is safe, while SMC does not have this ability, in general. Note that there is one case that SMC requires more simulations. It reveals one drawback of SMC.

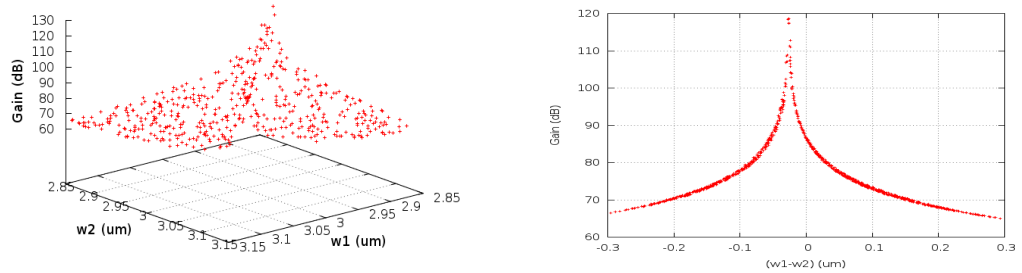
TABLE III: A list of opamp properties and the types of simulation used for measurement.

ID	PROPERTY	UNIT	SIMULATION	DESCRIPTION
1	Input offset voltage	mV	DC	DC input voltage that must be applied to cancel the DC offset within the opamp.
2	DC voltage gain	dB	AC	Small-signal voltage gain at DC.
3	Unity-gain bandwidth	MHz	AC	Frequency at which the small-signal gain first equals one.
4	Phase margin	$^\circ$	AC	Phase shift of the output voltage at the unity-gain bandwidth frequency.
5	Common-mode rejection ratio (CMRR)	dB	AC	Rejection to the change of the common-mode input voltage.
6	Positive power supply rejection ratio (PSRR$^+$)	dB	AC	Rejection to the change of the positive supply voltage.
7	Negative power supply rejection ratio (PSRR$^-$)	dB	AC	Rejection to the change of the negative supply voltage.
8	Positive slew rate (SR$^+$)	V/μs	Transient	Rate of change in the output voltage for a rising step change in the input.
9	Negative slew rate (SR$^-$)	V/μs	Transient	Rate of change in the output voltage for a falling step change in the input.



(a) w_1 and w_2 as parameters (b) $w_1 - w_2$ as parameter

Fig. 6: Relationship between unity-gain bandwidth, w_1 and w_2.



(a) w_1 and w_2 as parameters (b) $w_1 - w_2$ as parameter

Fig. 7: Relationship between DC voltage gain, w_1 and w_2.

When θ is close to the actual probability that the specification is satisfied, the convergence becomes slow and requires a large number of simulations. On the contrast, SMI is not affected in those cases because it performs model inference rather than hypothesis testing.

Finally, we note that the number of simulations is highly dependent on the number of parameters. The two properties with 8 parameters, the positive and the negative slew rate, require the largest number of simulations. It is clear that decreasing the number of parameters has a significant effect on reducing the number of simulations. One exception is the property DC voltage gain. In this case, a 4-piece polynomial is employed, in which each piece is generated from 60 simulations. Thus, a total of 240 simulations is performed.

Now we derive the set of parameters that are safe for the first two specifications. For visualization purpose, we fix the parameters $t_{ox1} - t_{ox2}$ and $v_{t01} - v_{t02}$ and search for the values of $w_1 - w_2$ and $l_1 - l_2$ that satisfy each specification. Figure 9 shows the results. Clearly, the intersection of the two shaded regions are safe for both of them.

V. Conclusion

In this paper, a statistical verification technique, statistical model inference, is proposed to handle the verification of analog circuits under process variation. SMI constructs statistically sound models that contain useful information about how process parameters affect design properties. They can be used to characterize a safe subset of the parameter values that satisfy all the design properties. Our contribution is to extend the ability of statistical verification from answering a yes/no question to model inference.

There are several possible directions for future work. First, we are interested in comparing the application of extreme value theory and statistical model checking in the context of analog circuit verification. Attempts will be made to combine the strengths of the two approaches. Second, we plan to integrate

TABLE V: Verification result of the opamp. Table III describes the property IDs. Note that each property may involve a different type of simulation and circuit configuration.

ID	SPEC	MONTE-CARLO			SMC		SMI				
		MEAN	STDDEV	YIELD	#SIMS	VERIFIED	#SIMS-1	TIME-1	#SIMS-2	TIME-2	VERIFIED
1	< 30 mV	24.4	17.4	65%	32, 6s	×	60, 10s	0.7s	187, 31s	0.2s	×, $> 57\%$
2	> 65 dB	75.6	9.7	96%	1104, 59s	×	240, 11s	3s	201, 10s	0.4s	×, $> 89\%$
3	> 5MHz	5.3	0.074	100%	89, 5s	✓	125, 6s	2s	111, 6s	0.1s	✓
4	$> 60^\circ$	64.1	0.35	100%	89, 5s	✓	125, 6s	2s	117, 6s	0.1s	✓
5	> 80dB	87.2	4.1	100%	89, 7s	✓	125, 9s	2s	134, 10s	0.1s	✓
6	> 80dB	90.7	2.3	100%	89, 7s	✓	125, 9s	2s	119, 9s	0.2s	✓
7	> 100dB	105.2	1.1	100%	89, 7s	✓	125, 9s	2s	98, 8s	0.1s	✓
8	$> 10\text{V}/\mu s$	10.7	0.08	100%	89, 31s	✓	250, 96s	5s	112, 41s	0.3s	✓
9	$< -7\text{V}/\mu s$	7.25	0.01	100%	89, 31s	✓	250, 99s	5s	139, 43s	0.3s	✓

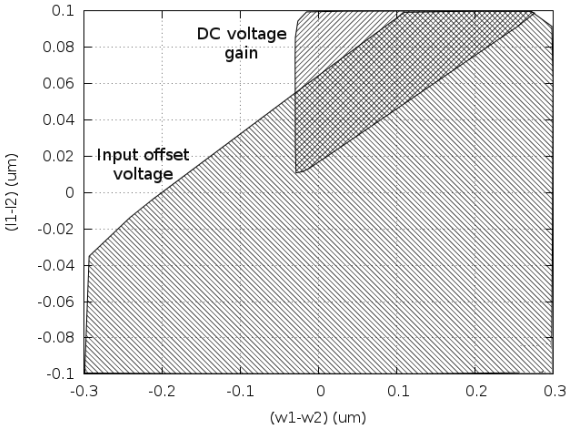


Fig. 9: The safe subsets of parameters for specification 1 and 2 with $t_{ox1} - t_{ox2}$ and $v_{t01} - v_{t02}$ fixed.

SMI with polynomial chaos expansion (PCE) technique which serves as a powerful alternative for least square fitting. Our preliminary research shows that PCE is a more efficient approach to generate polynomials according to a certain input distribution. Finally, it will be beneficial to explore machine learning techniques to automate the construction of piecewise polynomials when the underlying property is not well-behaved with respect to its input parameters.

REFERENCES

[1] Y. Zhang, S. Sankaranarayanan, and F. Somenzi, "Piecewise linear modeling of nonlinear devices for formal verication of analog circuits," in *FMCAD*, 2012, pp. 196–203.

[2] H. L. S. Younes and R. G. Simmons, "Probabilistic verification of discrete event systems using acceptance sampling," in *CAV*, 2002, pp. 223–235.

[3] S. K. Jha, E. M. Clarke, C. J. Langmead, A. Legay, A. Platzer, and P. Zuliani, "A Bayesian approach to model checking biological systems," in *CMSB*, 2009, pp. 218–234.

[4] P. Zuliani, A. Platzer, and E. M. Clarke, "Bayesian statistical model checking with application to Simulink/Stateflow verification," in *HSCC*, 2010, pp. 243–252.

[5] "Ngspice, an open-source circuit simulation." [Online]. Available: http://ngspice.sourceforge.net/

[6] R. P. Kurshan and K. L. McMillan, "Analysis of digital circuits through symbolic reduction," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 10, no. 11, pp. 1356–1371, 1991.

[7] L. Hedrich and E. Barke, "A formal approach to nonlinear analog circuit verification," in *ICCAD*, 1995, pp. 123–127.

[8] G. Frehse, B. H. Krogh, and R. A. Rutenbar, "Verifying analog oscillator circuits using forward/backward abstraction refinement," in *DATE*, 2006, pp. 257–262.

[9] S. Little, D. Walter, C. J. Myers, R. A. Thacker, S. Batchu, and T. Yoneda, "Verification of analog/mixed-signal circuits using labeled hybrid Petri nets," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 30, no. 4, pp. 617–630, 2011.

[10] D. Walter, S. Little, C. J. Myers, N. Seegmiller, and T. Yoneda, "Verification of analog/mixed-signal circuits using symbolic methods," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 27, no. 12, pp. 2223–2235, 2008.

[11] S. K. Tiwary, A. Gupta, J. R. Phillips, C. Pinello, and R. Zlatanovici, "First steps towards SAT-based formal analog verification," in *ICCAD*, 2009, pp. 1–8.

[12] M. Althoff, S. Yaldiz, A. Rajhans, X. Li, B. H. Krogh, and L. T. Pileggi, "Formal verification of phase-locked loops using reachability analysis and continuization," in *ICCAD*, 2011, pp. 659–666.

[13] L. Yin, Y. Deng, and P. Li, "Verifying dynamic properties of nonlinear mixed-signal circuits via efficient SMT-based techniques," in *ICCAD*, 2012, pp. 436–442.

[14] C. Yan and M. Greenstreet, "Oscillator verication with probability one," in *FMCAD*, 2012, pp. 165–172.

[15] A. Wald, "Sequential tests of statistical hypotheses," *The Annals of Mathematical Statistics*, vol. 16, no. 2, pp. 117–186, 1945.

[16] K. Sen, M. Viswanathan, and G. Agha, "Statistical model checking of black-box probabilistic systems," in *CAV*, 2004, pp. 202–215.

[17] H. Jefferys, "Some tests of significance, treated by the theory of probability," *Proc. Cambridge Philosophy Society*, no. 31, pp. 203–222, 1935.

[18] R. E. Kass and A. E. Raftery, "Bayes factors," *J. Amer. Stat. Assoc.*, vol. 90, no. 430, pp. 774–795, 1995.

[19] Y.-C. Wang, A. Komuravelli, P. Zuliani, and E. M. Clarke, "Analog circuit verification by statistical model checking," in *ASP-DAC*, 2011, pp. 1–6.

[20] A. Singhee and R. A. Rutenbar, *Novel Algorithms for Fast Statistical Analysis of Scaled Circuits*, ser. Lecture Notes in Electrical Engineering. Springer, 2009, vol. 46.

[21] D. Xiu, *Numerical Methods for Stochastic Computation: A Spectral Method Approach*. Princeton university Press, 2010.

[22] K. Strunz and Q. Su, "Stochastic formulation of spice-type electronic circuit simulation with polynomial chaos," *ACM Trans. Model. Comput. Simul.*, vol. 18, no. 4, 2008.

[23] H. Yoon, P. Variyam, A. Chatterjee, and N. Nagi, "Hierarchical statistical inference model for specification based testing of analog circuits," in *VLSI Test Symposium*, 1998, pp. 145–150.

[24] T. Dang and T. Nahhal, "Coverage-guided test generation for continuous and hybrid systems," *Formal Methods in System Design*, vol. 34, no. 2, pp. 183–213, 2009.

[25] S. N. Ahmadyan, J. A. Kumar, and S. Vasudevan, "Goal-oriented stimulus generation for analog circuits," in *DAC*, 2012, pp. 1018–1023.

[26] P. E. Allen and D. R. Holberg, *CMOS Analog Circut Design*, 2nd ed. Oxford University Press, Incorporated, 2002.